Rapport de fin de stage

Protéger la vie privée via un réseau adversarial d'attaque de réidentification

Vincent Vilain
Stagiaire de recherche - Projet DEEL (DEpendable and Explainable Learning)

obvia

UNIVERSITÉ LAVAL

Août 2024

Remerciements

Je tiens à remercier tout particulièrement l'Obvia pour avoir fourni une bourse, ainsi que l'IID (Institut Intelligence et Données), dans le cadre du projet DEEL, pour avoir financé ce stage au sein du laboratoire.

Je tiens aussi à remercier Youcef Korichi et Alec Larbrisseau pour le travail qu'ils ont réalisé à mes côtés et le temps qu'ils ont su prendre pour répondre à mes questions.

Pour finir, je souhaite remercier les professeurs Nadia Tawbi, Sébastien Gambs et ma directrice de stage Josée Desharnais pour leur accueil au sein de leur équipe et leur encadrement tout au long de mon stage. Ils ont rendu ce stage possible et ont su m'aider à le réaliser.

Produit avec le soutien financier des Fonds de recherche du Québec

Fonds de recherche Québec

ISBN : 978-2-925138-57-0
https://doi.org/10.61737/TABE1427

Table des matières

Table des figures

Liste de symboles et abréviations

DEEL : DEpendable and Explainable Learning LSTM : Long Short-Term Memory

CNN : Convolutional Neural Netwok

GPT : Generative Pre-trained Transformer

BERT : Bidirectional Encoder Representations from Transformers

RoBERTa : Robustly Optimized BERT Pretraining Approach

GAN : Generative Adversarial Networks

TUL : Trajectoty-User Linking

Résumé

Avec la précision des systèmes actuels GPS, les services géolocalisés sont devenus très populaires. Cette utilisation de plus en plus importante génère une très grande quantité de données de mobilité. Ces données peuvent servir à la planification urbaine, pour les transports en commun ou le développement de routes. Elles peuvent aussi servir à cibler des publicités à leurs utilisateurs. Pour protéger la vie privée des utilisateurs de ces services géolocalisés, il est important de trouver une méthode permettant de parer les attaques menaçant la vie privée à travers l'accès à ces données sensibles. Pour cela, les travaux de recherche de mon équipe d'accueil visent à élaborer des mécanismes de défense basés sur les attaques adversariales, à savoir des mécanismes qui apprennent les défenses en utilisant l'apprentissage des attaques [11]. Une de ces attaques est l'attaque de réidentification des utilisateurs. Elle repose sur un deuxième modèle qui a pour but d'assigner un utilisateur à une trajectoire.

Notre projet a pour but de créer une librairie Python qui sera utilisée dans le projet en cours de développement par mon équipe. Cependant, nous l'avons conçu de façon modulaire sous la forme d'une librairie que nous avons appelée MobiDeel. Cela destine ce travail à une utilisation plus large permettant à d'autres chercheurs travaillant sur les données de mobilité et les mécanismes de défense de la vie privée de tester facilement leurs méthodes face à des attaques de réidentification. La librairie étant open-source, la communauté de chercheurs peut participer librement à son développement.

1. Introduction

1. Introduction

1.1 Motivation

1.1.1 Mise à disposition de la librairie

Le but de ce projet est de mettre à disposition de la communauté de chercheurs en vie privée une librairie Python qui leur permette, dans le cadre d'élaboration de mécanismes de défense, de rapidement et facilement mettre en place une attaque de réidentification sur un modèle d'obfuscation de données de mobilité.

Ce travail s'inscrit dans le cadre d'un projet mené par l'équipe qui m'a accueilli. Ce projet consiste à élaborer des techniques et des mécanismes visant la protection de la vie privée pour les données de mobilité [11]. Une manière de créer une nouvelle méthode d'obfuscation des données est d'utiliser un réseau de génération adversarial [8] (GAN). Le but est d'entraîner un algorithme d'apprentissage automatique en l'attaquant avec d'autres algorithmes. C'est-à-dire qu'un modèle va essayer d'obfusquer les données et d'autres modèles vont essayer d'attaquer cette obfuscation afin de compromettre la vie privée des utilisateurs. La qualité de la défense dépend donc en partie de la qualité des attaques qui lui sont faites. C'est dans ce cadre que nous avons créé un modèle au niveau de l'état de l'art actuel en termes d'attaques de réidentification. Ce projet est une brique essentielle servant à la création de nouvelles méthodes d'obfuscation dont la finalité est la protection de la vie privée des utilisateurs. Et ce, tout en conservant l'utilité et la finalité pour lesquelles ces données ont été récoltées. Ce travail sert donc à réaliser le mécanisme de défense basé sur le réseau adversarial en cours de réalisation par l'équipe d'accueil. Ce projet est décrit dans un article accepté dans la conférence ESORICS 2024 [11].

Notre réalisation consiste à concevoir et implémenter une libraire que nous avons appelée MobiDeel. Grâce à la librairie MobiDeel, les chercheurs en vie privée pourraient tester facilement de nouvelles méthodes d'obfuscation et mesurer leur efficacité en les attaquant avec l'état de l'art actuel de la réidentification. Il serait aussi possible d'utiliser les modèles d'attaque pour mieux entraîner les modèles d'obfuscation en leur permettant d'avoir un retour pendant leur entraînement. C'est toute la stratégie de régularisation adversariale.

La librairie a été créée de manière à être la plus flexible possible afin de permettre à d'autres membres de la communauté de participer à son développement en ajoutant facilement de nouvelles méthodes de réidentification de trajectoires.

2. Géolocalisation et respect de la vie privée

2. Géolocalisation et respect de la vie privée

Les données de mobilité représentent les déplacements d'un utilisateur sur une période donnée. Elles peuvent être collectées de nombreuses manières différentes et parfois sans même que l'utilisateur soit au courant. De plus, les données de mobilité sont en majorité collectées par des acteurs privés (fournisseurs de service de location, opérateurs mobiles, etc.), mais aussi des institutions publiques. Elles sont souvent anonymisées en retirant simplement l'identifiant de l'utilisateur. Cependant, si ces données deviennent publiques, que ce soit par souci de transparence, une fuite de données ou autre, s'il est possible de réidentifier un utilisateur, sa vie privée est affectée. Ainsi, les données de mobilité peuvent révéler, par une analyse de point d'intérêts par exemple, des informations très personnelles comme l'état de santé, une appartenance politique ou religieuse, etc.

2.1 Équipe de travail

2.1.1 Projet DEEL

Notre projet s'inscrit dans le cadre du projet DEEL (DEpendable and Explainable Learning) qui se concentre sur l'application d'algorithmes de machine learning sur les données de transport et de mobilité. Le but de ce projet est de faire collaborer des partenaires académiques et industriels pour développer des intelligences artificielles interprétables, robustes, sécuritaires et certifiables. Mon stage fait partie de l'effort de protection de la vie privée, un des axes du projet DEEL. En outre, cet effort figure parmi les axes de travail du pôle d'expertise en cybersécurité et impacts sociétaux de l'Obvia.

2.1.2 Composition de l'équipe

Notre équipe de développement était composée de Youcef Korichi, étudiant au doctorat, Alec Larbrisseau, stagiaire étudiant au cégep et moi-même, Vincent Vilain, étudiant au baccalauréat. Nous étions supervisés par les professeurs Josée Desharnais, spécialiste en vérification formelle et cybersécurité, Sébastien Gambs, spécialiste en protection de la vie privée et enjeux éthiques de l'apprentissage machine, et Nadia Tawbi, spécialiste en cybersécurité.

2.2 État de l'art

2.2.1 Utilisation de l'architecture transformeur

L'architecture transformeur [17] repose sur un mécanisme d'attention. Ce mécanisme permet de donner une plus grande importance à certaines parties des données qui lui sont injectées. De cette manière, un modèle d'apprentissage profond peut prendre en compte le contexte qui entoure certaines sections des données pour mieux les comprendre et trouver plus efficacement des relations à l'intérieur des données. Par exemple, dans la phrase « Le chat mange sa nourriture parce qu'il a faim. », un modèle utilisant l'architecture transformeur prend en compte le contexte global de la phrase lorsqu'il analyse le mot « il », ce qui lui permet de comprendre que « il » fait référence à « le chat ».

De plus, l'architecture transformeur permet un traitement en parallèle des données. Cette méthode de traitement est bien plus rapide que le traitement séquentiel utilisé par les architectures précédentes. Cette architecture est utilisée par les modèles actuels les plus connus comme les ChatGPT fait par OpenAI [2] et bien d'autres. Sa polyvalence nous permet de l'utiliser pour réaliser l'attaque de réidentification, mais elle est aussi utilisée pour d'autres attaques comme l'extraction d'emplacements sensibles [21], le profilage et la prédiction de comportement [16]. Cette architecture est donc déjà grandement utilisée dans le domaine de la protection de la vie privée sur les données de mobilité, et c'est cette architecture qui a été adoptée au sein de mon équipe d'accueil [11].

2.2.2 Méthodes d'attaque adversarial

Les méthodes d'obfuscation reposant sur le GAN utilisent différentes méthodes d'attaques pour améliorer leurs performances. Par exemple, les attaques d'appartenance, les attaques d'homogénéité, etc. Nous avons réalisé une méthode d'association utilisateur-trajectoire (TUL) et il existe déjà de nombreuses façons de le faire. Comme décrit dans l'article [11], des techniques reposants sur les RNN, les Autoencodeurs Variationnels (VAE), les Réseaux Adversariaux Génératifs (GAN) et les mécanismes d'attention ont été exploitées pour améliorer les performances et capturer des motifs complexes. Cependant, les RNN standard montrent des limitations pour traiter de larges fenêtres contextuelles, essentielles pour comprendre les dépendances à long terme dans les trajectoires humaines. De nouvelles approches, comme l'apprentissage par distillation mutuelle ou l'utilisation des réseaux siamois, cherchent à améliorer ces performances, notamment pour des données de mobilité peu denses. Le modèle proposé dans ce contexte utilise l'architecture des transformateurs pour extraire efficacement des caractéristiques riches des trajectoires, ce qui est essentiel pour les attaques adversariales en TUL, tout en réduisant la complexité de l'extraction manuelle des caractéristiques et en permettant une mise en œuvre évolutive du processus d'apprentissage machine.

2.3 Problématique

Les données de trajectoires étant composées d'une suite de positions avec un horodatage pour chaque position, elles sont étroitement liées aux habitudes des utilisateurs. Ce projet avait pour but de réaliser une nouvelle approche pour assigner un utilisateur déjà connu à des trajectoires anonymisées. Pour cela, nous avons appliqué l'état de l'art actuel en termes de traitement de langage naturel (NLP) sur les données de trajectoires disponibles afin de montrer qu'il était possible de classifier des trajectoires en suivant ces approches. Plus précisément, l'objectif est d'assigner une trajectoire à un utilisateur dont certaines trajectoires sont déjà connues. Cette nouvelle méthode devait largement surpasser les méthodes manuelles qui sont très chronophages et même surpasser l'état de l'art actuel en termes de réidentification à des fins d'attaque adversariales.

2.4 Défis

2.4.1 Disponibilité des ensembles de données de mobilité

Les données nécessaires pour notre analyse doivent comporter des utilisateurs avec des trajectoires composées de positions en coordonnées GPS et de l'horodatage de ces positions. Les trajectoires peuvent être composées de nombres très différents de positions et donc varier de manière significative en longueur. Ainsi, certaines sont composées de très peu de points, ce qui peut poser des problèmes dans notre analyse, comme la baisse de la précision. En effet, si les points sont trop écartés, il est plus difficile de réidentifier un utilisateur parmi eux. De plus, une trajectoire trop longue, avec plus de 500 positions, ne peut pas être traitée directement par notre modèle et doit être soit découpée, soit échantillonnée pour qu'elle puisse être traitée.

La figure 1 montre les deux cas, quand la trajectoire a trop peu de points pour que le modèle puisse en tirer des informations concluantes et quand la trajectoire comporte trop de points, car la trajectoire est trop longue pour que le modèle puisse l'analyser en entier.

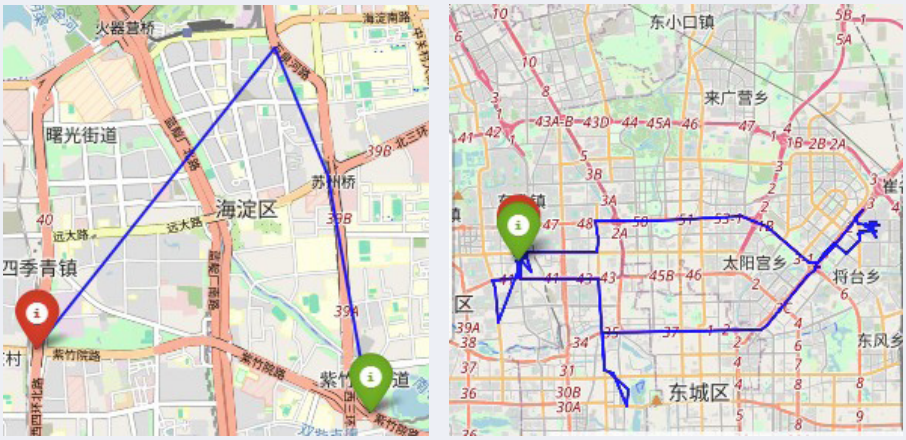

FIGURE 1 – Une trajectoire trop courte et trop longue pour le modèle, respectivement à gauche et à droite.

Les données de locations aussi précises sont extrêmement sensibles et sont donc très rares dans des jeux de données publiques. En effet, les entreprises collectant ce genre de données préfèrent les traiter elles-mêmes et vendre leurs résultats. De plus, depuis quelques années, ces données privées ne peuvent plus être partagées librement dans de nombreux pays en raison de nouvelles lois sur la protection des données et le respect de la vie privée.

Pour notre projet, nous devons donc nous baser sur les jeux de données disponibles qui sont créés dans le cadre d'études rendues publiques. Ces jeux de données sont malheureusement peu nombreux et ce manque de données réduit notre précision du fait que notre approche donne de meilleurs résultats quand elle peut apprendre avec un plus grand nombre de données.

Nous avons identifié six jeux de données utilisables dans notre projet :

- Le jeu de données Geolife [22]. Il a été collecté par Microsoft Research sur des employés de Microsoft Asie sur plus de quatre ans (d'avril 2007 à octobre 2011). Il comprend 17 621 trajectoires GPS représentant un total de 1 251 654 kilomètres. Plus de 90% de ces trajectoires sont constituées de positions collectées toutes les 1 à 5 secondes ou tous les 5 à 10 mètres. Ces trajectoires représentent une grande variété de déplacements extérieurs des utilisateurs comme des trajets quotidiens (travail, maison) ainsi que des activités de loisirs (shopping, tourisme, randonnée, vélo). Ces données sont concentrées dans le centre-ville de Pékin en Chine. La figure 2 représente les positions qui composent les trajectoires dans la ville de Pékin.

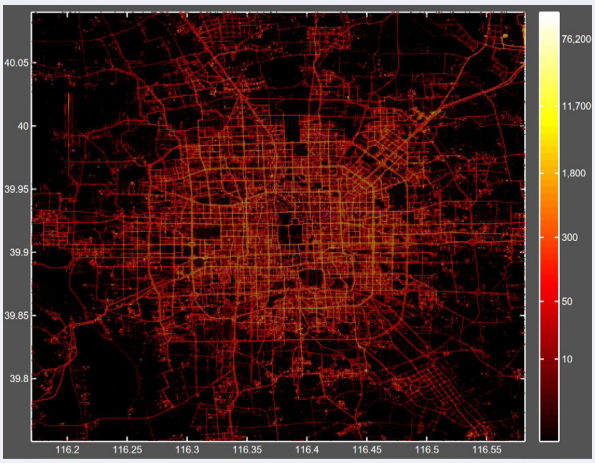

FIGURE 2 – Visualisation des positions des trajectoires du jeu de données Geolife avec la couleur qui indique la densité de points à cet endroit.[1]

- Le jeu de données T-Drive [20]. Il a été collecté par Microsoft Research sur une semaine, du 2 février au 8 février 2008, à Pékin. Il comprend les trajectoires GPS de 10 357 taxis, totalisant environ 15 millions de points GPS pour une distance totale de 9 millions de kilomètres. Les positions consécutives sont espacées en moyenne d'environ 177 secondes et 623 mètres. Ces trajectoires couvrent une variété de déplacements urbains, avec une densité de points plus élevée dans certaines zones. Ainsi, ces données offrent une vue détaillée des déplacements des taxis dans Pékin.

- Le jeu de données Gowalla [4]. Il a été collecté par une équipe de recherche universitaire sur la période de février 2009 à octobre 2010. Ce jeu de données comprend les trajets GPS de 5000 utilisateurs à travers plusieurs villes européennes. Ces utilisateurs utilisaient le réseau géosocial Gowalla et partageaient eux-mêmes leur position. Le jeu de données Gowalla totalise environ 25 millions de points GPS pour une distance totale de 12 millions de kilomètres. L'intervalle moyen d'échantillonnage est d'environ 120 secondes et 500 mètres entre deux points consécutifs. Cette densité de localisations est bien inférieure à celle des jeux de données précédents. Ces trajectoires représentent une diversité de déplacements quotidiens et occasionnels, incluant les trajets domicile-travail, les sorties de loisirs et les activités sportives. Les données montrent une concentration plus élevée de points GPS dans les centres-villes et les zones touristiques, offrant ainsi une vue détaillée des schémas de mobilité urbaine en Europe.

- Le jeu de données Foursquare New York [19]. Il a été collecté par les utilisateurs du réseau géosocial Foursquare sur une période d'un an, de janvier à décembre 2017. Ce jeu de données comprend les check-ins GPS de milliers d'utilisateurs à travers la ville de New York. Il totalise environ 10 millions de points de check-in, représentant une diversité d'activités telles que les visites de restaurants, les sorties nocturnes, les visites touristiques et les trajets quotidiens. Les données montrent une concentration élevée de points dans les quartiers populaires comme Manhattan et Brooklyn. Cependant, chaque trajectoire comporte un nombre assez faible de positions.

1 https://www.microsoft.com/en-us/research/publication/geolife-gps-trajectory- dataset-user-guide

- Le jeu de données Foursquare Tokyo [19]. Collecté sur la même période que le jeu de données de New York (janvier à décembre 2017), ce jeu de données comprend les check-ins GPS de milliers d'utilisateurs dans la ville de Tokyo. Il totalise environ 8 millions de points de check-in, couvrant une variété d'activités allant des restaurants et cafés aux centres commerciaux et attractions touristiques. Comme pour celui de New York, ce jeu de données à une densité de localisations par trajectoire beaucoup plus faible que les jeux de données Geolife ou T-Drive. Les données montrent une concentration élevée dans certains quartiers, fournissant ainsi une vue détaillée des schémas de mobilité et des points d'intérêt les plus populaires à Tokyo.
- Le jeu de données Brightkite [3]. Il a été collecté par les utilisateurs du réseau géosocial Brightkite sur une période de deux ans, de mars 2008 à avril 2010. Ce jeu de données comprend les enregistrements GPS de milliers d'utilisateurs à travers le monde. Il totalise environ 4,5 millions de points d'enregistrement, couvrant une large variété de lieux et d'activités tels que les restaurants, les magasins, les parcs et les lieux de travail. Les données montrent une concentration élevée de points dans les grandes villes et les zones urbaines, offrant une vue globale des comportements de mobilité des utilisateurs de Brightkite.

2.4.2 Forte capacité de calcul

L'entraînement de modèles avec les architectures que nous utilisons demande une grande quantité de données, mais aussi des capacités de calcul importantes afin de pouvoir les traiter dans un délai raisonnable. Nous avons donc dû faire des choix sur la taille de notre modèle pour pouvoir réaliser notre projet avec les ressources à notre disposition.

2.5 Objectifs de réidentification

Nous nous attendions à ce que les modèles produits puissent assez précisément assigner des trajectoires à des utilisateurs. En effet, nous avons choisi une architecture créée pour le type de tâche que nous réalisons et nous avons adapté nos données de manière à les rendre compatibles avec cette architecture. De plus, en raison de leur nature, nos données possèdent des régularités internes que notre architecture est spécifiquement conçue pour déceler. Le seul problème qu'il nous restait était la quantité de données disponibles et la puissance de calcul à notre disposition.

3. Méthodologie

3. Méthodologie

3.1 Prétraitement des données

Nous avons commencé par analyser les différents jeux de données qui étaient à notre disposition pour se familiariser avec leur format et identifier les étapes de prétraitement nécessaires. Pour cela, nous avons réalisé des analyses statistiques sur les données et créé un module pour représenter ces données afin de les visualiser facilement.

Les données que nous récupérons dans les jeux de données publiques sont organisées dans des formats parfois très différents. Nous devons donc les prétraiter afin qu'elles aient toutes le même format et qu'elles soient compatibles avec notre architecture de modèle. Étant donné que le format d'origine varie selon le jeu de données, nous laissons la possibilité aux utilisateurs de notre librairie de définir eux-mêmes leur manière d'importer leurs données. De plus, le format des données devait être compatible avec les méthodes d'entraînement de modèle que nous avons utilisées.

Cette étape de prétraitement influence énormément les performances des modèles. Il est donc important de ne pas détériorer les données en réalisant des opérations inadéquates. Autrement, un modèle ne pourra pas correctement s'entraîner sur les données ou les résultats seront invalides.

3.1.1 Suppression des valeurs extrêmes

Pour supprimer les valeurs extrêmes et donc améliorer la qualité de nos données, nous avons décidé de réutiliser certaines parties de nos analyses statistiques pour éliminer toutes les positions qui se trouvaient hors d'un certain rayon du centre de la ville. En effet, certains points de nos jeux de données se trouvent aux coordonnées GPS (0, 0) sûrement à cause d'une erreur du système GPS faisant la mesure. Nous avons aussi remarqué dans nos analyses que certaines trajectoires comprenaient des points hors de la ville que nous voulions analyser. Nous avons donc décidé de retirer ces données afin d'améliorer notre précision en nous focalisant sur une ville et donc une zone géographique précise.

3.1.2 Découpage des trajectoires

Dans les jeux de données, les positions sont simplement données sous forme de listes comprenant, à chaque ligne, une latitude, une longitude et un horodatage, et ce pour chaque utilisateur. Nous découpons donc cette très grande "trajectoire" en plusieurs sous-trajectoires. Nous avons implémenté plusieurs méthodes pour découper les données : par seconde, minute, heure, jour, semaine et mois. Nous utilisons principalement le découpage par jour, car les trajets quotidiens représentent bien les habitudes des différents utilisateurs dans nos jeux de données. Cependant, nous laissons la liberté aux utilisateurs de notre librairie de choisir ce qui leur convient le mieux et de ré-implémenter, si nécessaire, une méthode spéciale pour leurs besoins. En effet, si le découpage des données en trajectoire n'est pas bien fait, il se peut que certaines trajectoires soient coupées de manière inappropriée pendant cette étape du pré-traitement. Par exemple, si une trajectoire se déroule de 22h00 à 1h00 du matin et que la trajectoire est découpée par journées, la trajectoire sera coupée en deux trajectoires différentes avec l'une allant de 22h00 à 23h59 et l'autre de 00h00 à 1h00. Ceci dégraderait la qualité des données en modifiant les trajectoires des utilisateurs.

3.1.3 Filtrage des trajectoires

Les trajectoires peuvent être filtrées pour isoler certaines parties du jeu de données dans le but de garder seulement les données qui permettraient le mieux d'identifier chaque utilisateur. Cela permet aussi de réduire la taille du jeu de données pour se concentrer sur une sous-section de celui-ci. Par exemple, nous avons implémenté des filtres pour garder seulement les trajectoires qui ont été enregistrées pendant la semaine en excluant les fins de semaine. On peut aussi isoler celles qui se trouvent dans un intervalle de temps, celles qui comportent plus qu'un certain nombre minimum de positions, celles de certains utilisateurs, etc. Nous utilisons principalement le filtre qui permet de sélectionner certains utilisateurs pour réduire significativement la taille de notre jeu de donnée afin d'itérer rapidement sur nos solutions avant de les appliquer sur l'ensemble du jeu de données. Les utilisateurs de la librairie peuvent aussi définir les filtres qui leur sont nécessaires.

3.1.4 Simplification des trajectoires

Même une fois découpées, les trajectoires de certains utilisateurs sont parfois trop longues pour être traitées par notre modèle. Il faut donc les simplifier en retirant certains points, tout en affectant le moins possible la forme de la trajectoire. Nous avons implémenté différentes méthodes pour résoudre ce problème, comme par exemple une sélection aléatoire des positions à garder et aussi différents algorithmes comme l'algorithme de *Douglas-Peucker* [7], l'algorithme *Error-Search* [14], l'algorithme *SP* [6], etc. Ces simplifications nous permettent de réduire la taille de nos données en les modifiant le moins possible. De cette manière nous pouvons réaliser nos entraînements de modèles plus rapidement et en utilisant moins de ressources de calcul.

FIGURE 3 – Comparaison d'une même trajectoire simplifiée par différents algorithmes de gauche à droite, la trajectoire originale, trajectoire simplifiée avec l'algorithme SP sur l'originale et trajectoire simplifiée avec l'algorithme DP sur l'originale.

On peut remarquer, dans la figure 3, que les trajectoires n'ont que très peu changé. Pourtant, la trajectoire originale était composée de 41 points alors que la trajectoire simplifiée avec l'algorithme SP est composée de 27 points et que celle simplifiée avec l'algorithme DP est composée de seulement 23 points. Ces algorithmes, comme tous ceux qui ont été implémentés, peuvent être paramétrés pour modifier leur taux de compression de la trajectoire. Si le taux de compression est trop élevé ou que l'algorithme utilisé ne convient pas aux données, la simplification des trajectoires peut modifier grandement les performances des modèles créés par la suite. De manière similaire aux autres étapes du prétraitement, la simplification des trajectoires peut être rapidement ré-implémentée pour être plus performante sur un autre jeu de données.

3.1.5 Formatage des positions

Un geohash [15] est une méthode d'encodage géographique qui transforme des coordonnées GPS (latitude et longitude) en une chaîne de caractères alphanumériques. Chaque caractère ajouté à la chaîne augmente la précision de la localisation. Par exemple, un geohash de précision faible (moins de caractères) représente une grande zone, tandis qu'un geohash plus long (plus de caractères) représente une zone plus petite et précise. Cette technique permet de simplifier et accélérer la manipulation de données de localisation en les représentant de manière compressée.

Nous utilisons des geohash avec 8 caractères pour représenter nos données GPS en format texte pour notre modèle. Cela nous permet d'utiliser des architectures créées principalement pour la génération ou la classification de texte comme les GPT (*generative pre-trained transformer*) de OpenAI. Les sujets du choix d'architecture et du niveau de précision des geohash seront abordés plus en détail dans la section 3.3. La figure 4 montre le changement de précision en passant de 6 à 7 caractères pour représenter le geohash.

FIGURE 4 – Division de la section représentée par un geohash quand le nombre de caractères augmente.

Cette étape peut grandement impacter les performances des modèles générés par la suite. Elle définit la précision maximale que les modèles vont utiliser.

3.2 Visualisation des données

La représentation des données de trajectoires sur une carte nous a grandement aidés tout au long du projet afin de se représenter les données. Tout d'abord, nous utilisions des représentations des données pour les visualiser afin de mieux les comprendre. Par la suite, cette visualisation nous a permis de vérifier le fonctionnement de nos algorithmes de prétraitement comme le nettoyage des données et la simplification des trajectoires. Pour finir, nous avons pu représenter les trajectoires pendant l'entraînement de nos modèles pour vérifier si la reconstruction des trajectoires était correcte. Voir figures 6, 7 et 8.

3.3 Conception du modèle de réidentification

3.3.1 Choix de l'utilisation d'un transformeur

Nous avons décidé de nous orienter vers de l'apprentissage profond pour faire face au nombre considérable de données que les méthodes plus classiques ne peuvent pas traiter dans un temps raisonnable. De plus, nous avons écarté les LSTM (*long short-term memory*) [9], car les données, sous forme de tokens, doivent être passées une par une au modèle. Cela réduit énormément la vitesse de traitement des données du fait qu'il n'y ait pas de parallélisation. Nous avons aussi écarté les CNN (*convolutional neural network*) [12] du fait que leur attention locale ne permet pas de capturer les corrélations, à l'intérieur des données, aussi loin que peut le faire un transformeur. Nous avons choisi l'architecture des transformeurs pour sa rapidité d'inférence et ses capacités à trouver des corrélations grâce à son contexte.

3.3.2 Choix de l'architecture

Notre but est de créer un nouveau modèle pour la réidentification à partir d'une architecture existante qui est fonctionnelle et utilisée couramment. Nous avons donc commencé par rechercher les architectures disponibles avec des implémentations documentées et fréquemment utilisées. Notre problème est un problème de classification dont le but est d'attribuer une trajectoire à un utilisateur (ainsi chaque utilisateur peut être vu comme une classe). Pour ce type de problème, l'architecture BERT (*bidirectional encoder representations from transformers*) [5] est meilleure qu'une architecture comme GPT qui est plus orientée vers la génération de données comme du texte.

Depuis la parution de l'architecture BERT, de nombreuses variations ont été proposées pour des usages différents. Nous avons choisi d'utiliser l'architecture RoBERTa (*robustly optimized BERT pretraining approach*) [13] qui est une version plus robuste et optimisée de l'architecture BERT.

Nous avons opté pour l'implémentation de Hugging Face [18], car elle offre une bonne documentation et la librairie transformers de Hugging Face est très répandue dans les domaines de la recherche et en production. Cette librairie nous permet de ne pas avoir à ré-implémenter nous-mêmes l'architecture de RoBERTa ce qui nous fait gagner beaucoup de temps.

Nous avons opté pour l'utilisation de geohashs pour encoder nos données GPS, car l'architecture que nous utilisons est principalement faite pour travailler sur des données sous forme de texte comme les geohashs plutôt que sur des nombres à virgules comme les coordonnées GPS.

3.3.3 Choix de la précision des geohashs

Les transformeurs utilisent un vocabulaire fixe pour représenter le jeu de données. Si ce vocabulaire est trop grand, le modèle est très long à entraîner et il aura de mauvaises performances. Au contraire, s'il est trop petit, soit toutes les données ne pourront pas être représentées, soit leur représentation ne sera pas assez précise.

Dans notre cas, la taille de notre vocabulaire varie en fonction de la précision choisie pour nos geohashs. Nous avons choisi une précision de 8 caractères pour que suffisamment des points soient regroupés afin de réduire le nombre total de geohash différents utilisés pour représenter nos données. Donc nos coordonnées GPS sous forme de latitude et longitude sont représentées par des chaînes de caractère de longueur 8.

3.3.4 Entraînement des modèles

Après avoir choisi notre architecture et avoir transformé nos données pour qu'elles puissent être interprétées par le modèle, nous avons pu commencer à faire des entraînements afin de tester les capacités de nos modèles. Nous avons enregistré les performances des modèles que nous avons entraînés et les modifications faites sur chaque paramètre pour suivre de manière fiable leur évolution. La figure 5 nous montre l'interface que nous utilisions. Cette interface est créée et rendue accessible à l'équipe grâce à l'outil Weight and Biases [1].

FIGURE 5 – Interface montrant l'évolution de la performance des modèles pendant les entraînements.

Pendant l'entraînement, le modèle utilise des trajectoires dont certaines parties sont cachées et le modèle essaie de déduire la position des points qui ont été cachés. Les figures 6, 7 et 8 permettent de visualiser ces déductions.

Comme nous pouvons le voir dans la figure 6, l'inférence du modèle (en vert) va légèrement trop loin par rapport à la trajectoire initiale (en rouge). On pourrait croire que ce résultat montre une mauvaise portion de l'entraînement, mais en réalité si le modèle avait réussi à déduire la position exacte qui avait été cachée (en bleu), cela voudrait dire que le modèle a appris les données d'entraînement au lieu de trouver des relations à l'intérieur des données.

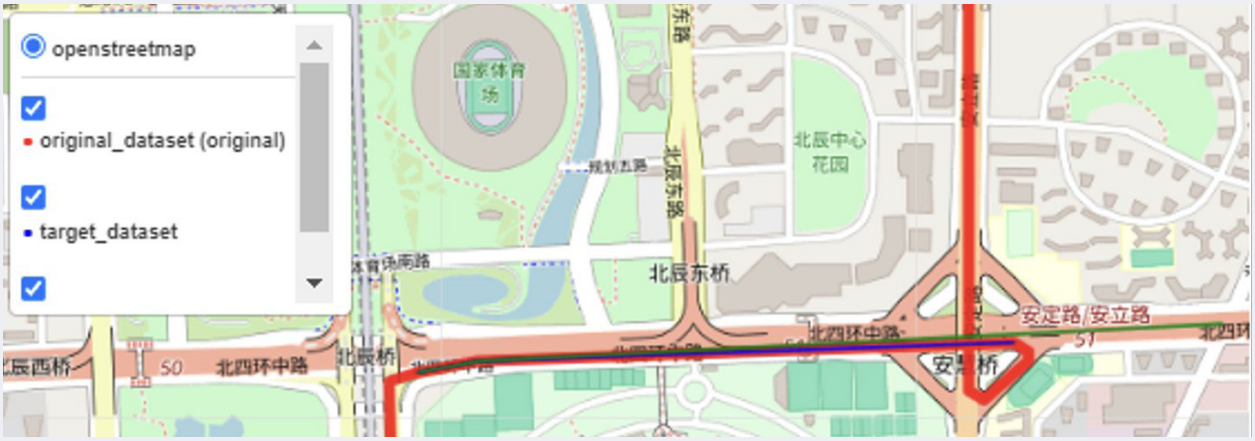

FIGURE 6 – Exemple de déduction pendant l'entraînement avec la trajectoire originale en rouge, la partie cachée en bleu et la partie déduite en vert.

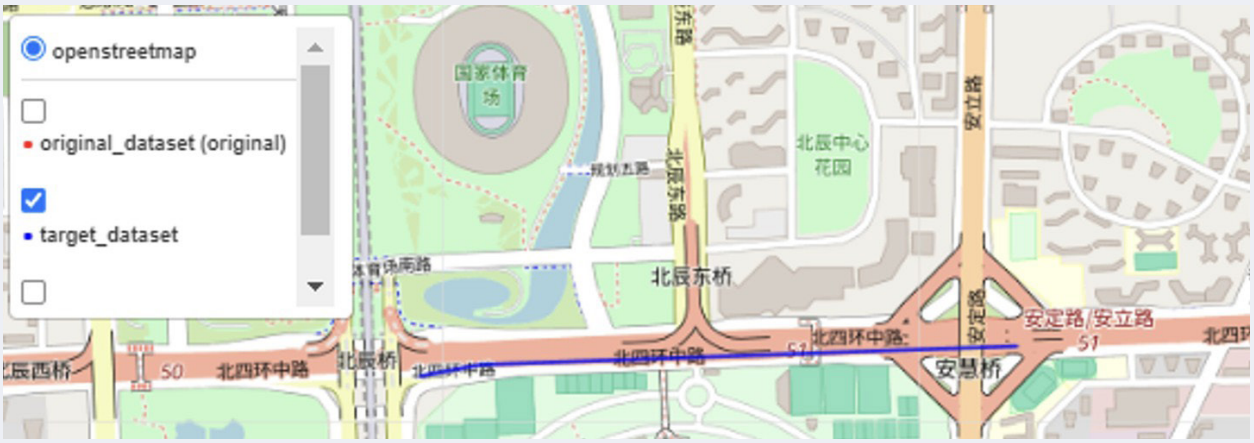

FIGURE 7 – Section de la trajectoire ayant été cachée et qui doit être déduite par le modèle pendant l'entraînement.

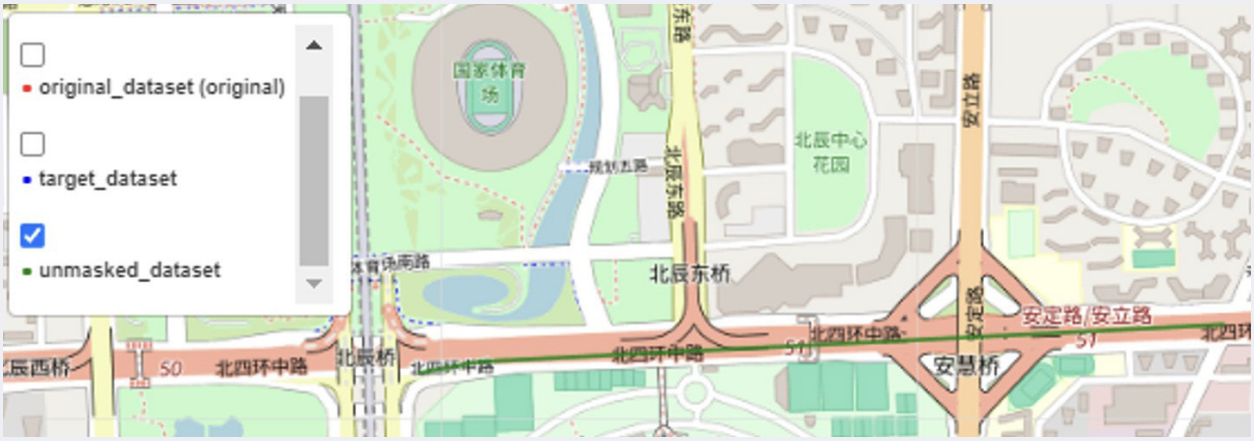

FIGURE 8 – Section de la trajectoire ayant été déduite par le modèle pendant l'entraînement.

3.3.5 Raffinage du modèle

Après avoir entraîné le modèle, il faut le raffiner pour lui indiquer quelle trajectoire correspond à quel utilisateur. Pour cela, nous avons divisé notre jeu de données en trois parties : la partie entraînement, la partie de test et la partie de la validation. La partie d'entraînement est la plus grande, elle contient les données ainsi que la classification qui est attendue. La partie validation sert au modèle à se tester pendant son entraînement pour vérifier qu'il arrive bien à classifier une trajectoire. Elle contient seulement les données sans la classification puisque c'est le modèle qui doit la trouver. Pendant le raffinage, le modèle fait plusieurs itérations sur ces données pour minimiser le taux d'erreur pendant le test. Enfin, la partie test sert à la fin de l'entraînement pour tester les performances du modèle et déterminer sa précision et son exactitude.

4. Résultats

4. Résultats

L'un des premiers résultats que nous avons eus est un pourcentage de confiance sur une classification faite par un modèle qui n'avait pas encore été raffiné. Nous avions demandé au modèle de déduire un utilisateur alors qu'à ce moment-là il ne s'était entraîné qu'à déduire des sections de trajectoires. À ce moment-là son taux de confiance pour l'association de la trajectoire à un utilisateur était d'environ 52 %. Le modèle répondait donc quasiment aléatoirement.

Nous avons aussi les performances du modèle pendant son entraînement lors de l'étape de raffinement. Comme le montre la figure 5, dans le graphe en haut à gauche, la perte du modèle diminue très bien pendant l'entraînement. La perte est une mesure de l'erreur entre les prédictions du modèle et les réponses correctes attendues.

5. Conclusion

5. Conclusion

Pendant ce stage, nous avons énormément fait progresser la librairie MobiDeel en rajoutant des fonctionnalités clés et en faisant de nombreuses améliorations. La figure 9 représente l'architecture logicielle de haut niveau que nous avons implémenté. Tout ce travail nous a permis d'entraîner des modèles pour classifier les trajectoires et les ré-assigner à un utilisateur. Nous avons réalisé tout cela en donnant le plus de liberté possible à de futurs utilisateurs de notre librairie pour qu'ils puissent l'adapter le plus possible à leurs données et leurs objectifs. Ce travail est en cours d'intégration dans le projet mené par mon équipe d'accueil et dont la finalité est la protection de la vie privée pour les données de mobilité [11].

Il ne manque plus que quelques étapes pour que la librairie soit accessible publiquement. Elle pourra alors aider les chercheurs à trouver une méthode d'obfuscation des données de mobilité qui puisse résister aux attaques de réidentification tout en conservant l'utilité des données.

6. Axes de progression

6. Axes de progression

6.1 Prise en compte de l'horodatage

Pour l'instant, nous nous sommes concentrés sur la séquence de position pour faire notre classification. Nous avons fait ce choix pour tester en un premier temps les performances atteignables avec seulement les positions. Prendre en compte directement le temps en plus de la position nous aurait sûrement empêchés d'aborder d'autres points que nous avons traités.

Le principal défi lié à l'inclusion de l'horodatage dans notre modèle est la manière d'encoder le temps et de l'ajouter à la position avant de donner le résultat au modèle. Par exemple, nous pourrions utiliser Time2Vec [10] qui permet d'encoder le temps même avec des fréquences temporelles variées.

Cette prise en compte du temps permettrait sûrement d'améliorer les performances du modèle, lui donnant plus de données de contexte permettant à l'algorithme de trouver des corrélations à l'intérieur des données.

6.2 Génération de données

Notre projet reposait sur la classification d'utilisateur à des fins de réidentification. Cependant, avec la flexibilité offerte par l'architecture transformeur, il est possible d'adapter notre modèle de classification pour faire de la génération de données synthétiques.

La génération de données synthétiques est une méthode très efficace pour anonymiser des données tout en conservant leur utilité. Pour cela, il faut un modèle ayant été entraîné sur des données similaires, et faire de la génération de données avec celui-ci. Le défi de la génération de données synthétiques est que ces données doivent ressembler suffisamment à de vraies données pour qu'elles puissent être utilisées, mais qu'elles ne doivent pas trop ressembler aux données d'entraînement sinon la protection de la vie privée est compromise.

6.3 Utilisation d'autres architectures pour le modèle

Grâce à notre architecture modulaire, la librairie peut facilement être adaptée pour faire fonctionner d'autres architectures que celle de RoBERTa que nous avons choisie. Tous les utilisateurs de notre librairie peuvent donc l'adapter pour fonctionner au mieux avec leurs données et leurs objectifs.

7. Annexe

7. Annexe

7.1 Diagramme haut niveau de MobiDeel

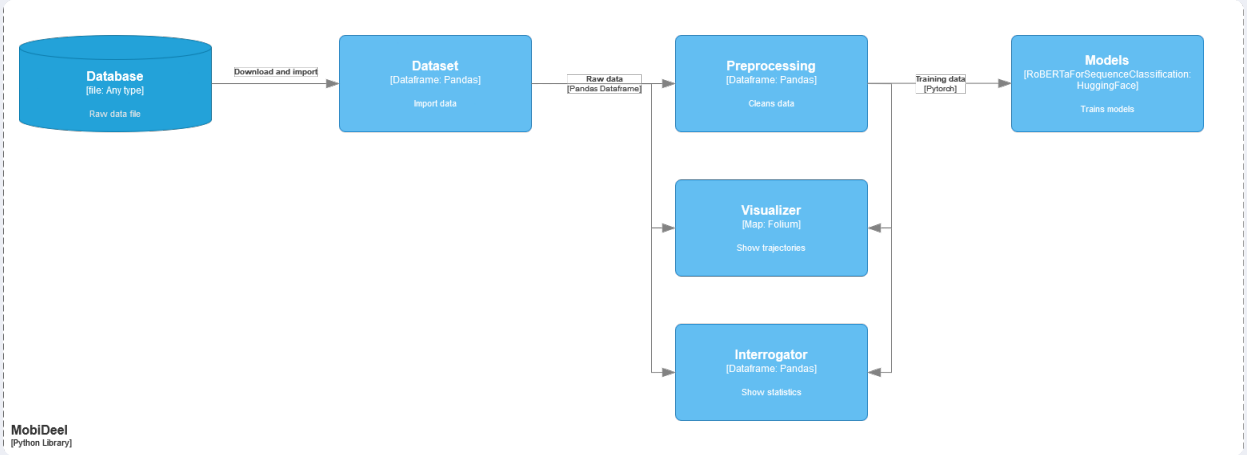

FIGURE 9 – Diagramme haut niveau de MobiDeel présentant les principales composantes de la librairie MobiDeel.

Références

[1] Lukas Biewald. *Experiment Tracking with Weights and Biases*. Software available from wandb.com. 2020. url : https://www.wandb.com/.

[2] Tom Brown et al. « Language models are few-shot learners ». In : *Advances in neural information processing systems* 33 (2020), p. 1877-1901.

[3] Eunjoon Cho, Seth A Myers et Jure Leskovec. *Brightkite Check-in Data*. Stanford Network Analysis Project (SNAP). 2011. url : https://snap.stanford.edu/data/ loc-brightkite.html.

[4] Eunjoon Cho, Seth A Myers et Jure Leskovec. « Friendship and Mobility : User Movement in Location-Based Social Networks ». In : *Proceedings of the 17th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. New York, NY, USA : ACM, 2011, p. 1082-1090. doi : 10.1145/2020408.2020579.

[5] Jacob Devlin, Ming-Wei Chang, Kenton Lee et Kristina Toutanova. « BERT : Pre-training of Deep Bidirectional Transformers for Language Understanding ». In : *North American Chapter of the Association for Computational Linguistics*. 2019. url : https://api.semanticscholar.org/CorpusID:52967399.

[6] Edsger W Dijkstra. « A note on two problems in connexion with graphs ». In : *Numerische mathematik* 1.1 (1959), p. 269-271.

[7] David H Douglas et Thomas K Peucker. « Algorithms for the reduction of the number of points required to represent a digitized line or its caricature ». In : *Cartographica : The International Journal for Geographic Information and Geovisualization* 10.2 (1973), p. 112-122.

[8] Qiang Gao et al. « Adversarial Mobility Learning for Human Trajectory Classification ». In : *IEEE Access* 8 (2020), p. 20563-20576. doi : 10 . 1109 / ACCESS. 2020 . 2968935. url : https://doi.org/10.1109/ACCESS.2020.2968935.

[9] Sepp Hochreiter et Jürgen Schmidhuber. « Long Short-Term Memory ». In : *Neural Computation* 9.8 (1997), p. 1735-1780. doi : 10.1162/neco.1997.9.8.1735.

[10] Seyed Mehran Kazemi et al. « Time2Vec : Learning a Vector Representation of Time ». In : *CoRR* abs/1907.05321 (2019). arXiv : 1907.05321. url : http://arxiv.org/ abs/1907.05321.

[11] Youcef Korichi, Josée Desharnais, Sébastien Gambs et Nadia Tawbi. « Leveraging Transformer Architecture for Effective Trajectory-User Linking (TUL) Attack Mitigation ». In : *Proceedings of the 29th European Symposium on Research in Computer Security (ESORICS)*. 2024.

[12] Yann Lecun, Leon Bottou, Yoshua Bengio et Patrick Haffner. « Geo ». In : *Proceedings of the IEEE* 86.11 (1998), p. 2278-2324. doi : 10.1109/5.726791.

[13] Yinhan Liu et al. *RoBERTa : A Robustly Optimized BERT Pretraining Approach*. 2019. arXiv : 1907.11692 [cs.CL]. url : https://arxiv.org/abs/1907.11692.

[14] Cheng Long, Raymond Chi-Wing Wong et H V Jagadish. « Trajectory simplification : On minimizing the direction-based error ». In : *Proceedings of the VLDB Endowment* 8.1 (2014), p. 49-60.

[15] Gustavo Niemeyer. « Geospatial coordinate encoding and compression system ». US20080157844A1. Juill. 2008. url : https : / / patents . google . com / patent / US20080157844A1/en.

[16] Roberto Pellungrini, Luca Pappalardo, Francesca Pratesi et Anna Monreale. « A data mining approach to assess privacy risk in human mobility data ». In : *ACM Transactions on Intelligent Systems and Technology (TIST)* 9.3 (2017), p. 1-27.

[17] Ashish Vaswani et al. « Attention is All You Need ». In : *Advances in Neural Information Processing Systems*. 2017, p. 5998-6008.

[18] Thomas Wolf et al. *HuggingFace's Transformers : State-of-the-art Natural Language Processing*. 2020. arXiv : 1910.03771 [cs.CL]. url : https://arxiv.org/abs/1910.03771.

[19] D Yang et al. *Nationwide check-in dataset*. Data provided by Foursquare for research purposes. 2014. url : https://www.kaggle.com/chetanism/foursquare-nyc-and- tokyo-checkin-dataset.

[20] Jing Yuan, Yu Zheng et Xing Xie. *T-Drive Trajectory Data*. Microsoft Research Asia. 2011. url : https://www.microsoft.com/en-us/research/publication/t- drive-trajectory-data-sample/.

[21] Qingchuan Zhao, Chaoshun Zuo, Giancarlo Pellegrino et Li Zhiqiang. *Geolocating Drivers : A Study of Sensitive Data Leakage in Ride-Hailing Services*. 2019.

[22] Yu Zheng, Qiang Li, Yukun Chen, Xing Xie et Wei-Ying Ma. *GeoLife GPS Trajectories*. Microsoft Research Asia. 2009. url : https:// www . microsoft. com / en - us / research / project / geolife - building - social - networks - using - human - location-history/.

obvia

obvia.ca